PRESTO单脉冲搜索的算法优化及其并行化*

傅志明[1]，陈宗浩[1]，梁楠[1]，于徐红[1]，邵广盛[2]，张彬[1]，谢晓尧[1*]
(1、贵州师范大学贵州省信息与计算科学重点实验室，贵州 贵阳 550001；
2、贵州师范大学图书馆，贵州 贵阳 550001)

摘 要：快速射电暴(Fast Radio Burst, FRB)及旋转射电暂现源(Rotating Radio Transient, RRAT)等一些没有明显周期性的天体发现以来，单脉冲搜索受到研究人员广泛关注，同时随着射电望远镜设备日趋完善，更高分辨率和更广阔的观测空间产生的观测数据量剧增，加快对观测数据的处理迫在眉睫。介绍了PRESTO (PulsaR Exploration and Search Toolkit)中单脉冲搜索，使用Cython编程方式对单脉冲搜索中去趋势算法进行优化，并通过Ray框架实现单脉冲搜索在中央处理器(Central Processing Units, CPU)上并行化。实验结果表明，算法优化后的单脉冲搜索并行化，能明显提升搜索程序性能，显著缩短数据处理时间，同时该并行策略仅基于中央处理器，无需修改代码即可在纯中央处理器环境下实现高性能数据处理。

关键词：单脉冲搜索；脉冲星；并行化
中图分类号：P111　　**文献标识码：**A　　**文章编号：** 1672-7673(2023)

1 介 绍

脉冲星是一类特殊的中子星，具有极强的磁场和极其稳定的旋转周期。基于脉冲星稳定的周期性，目前脉冲星探测主要使用快速傅里叶变换(Fast Fourier Transform，FFT)和快速折叠算法(Fast Folding algorithm，FFA)[1]在时域上周期性搜索。然而，探测研究显示，有一些脉冲星并没有呈现明显的周期性，如零脉冲星[2-3]、间歇性脉冲星，以及文[4]发现的特殊天文脉冲信号快速射电暴与文[5]发现的旋转射电暂现源，这类天体不能使用周期性搜索方法搜寻，需要通过单脉冲搜索进行搜寻。

进入21世纪以来，射电望远镜设备日趋完善，更高分辨率的观测使得脉冲星搜索数据量正以TB，PB甚至EB量级的速度快速增长[6]。2020年1月11日，500m口径球面射电望远镜(Five-hundred-meter Aperture Spherical radio Telescope，FAST)正式投入运行。随着FAST的高效运行，据估计，通过L波段19波束接收机收集的观测数据，每小时可达1.6TB，每年数据量在10~20 PB[7]，数据量的激增给存储和处理数据带来严峻考验。另外根据预测，银河系内大约有15万颗潜在脉冲星，其中可探测脉冲星大约有30000颗[8]。然而，目前在ATNF(Australia Telescope National Facility)收录的脉冲星数量不超过3500颗（https://www.atnf.csiro.au/research/pulsar/psrcat/）。因此，银河系中仍有许多脉冲星未探测到。高效处理这些数据已经极为迫切，否则会制约FAST对脉冲星的科学发现。

目前常用的单脉冲搜索工具有PRESTO(PulsaR Exploration and Search Toolkit)(https://www.cv.nrao.edu/~sransom/presto/)中single_pulse_search.py，SIGPROC(https://sourceforge.net/projects/sigproc/)中SEEK，以及基于图形处理器 (Graphics Processing Unit，GPU)加速的单脉冲搜索专用软件HEIMDALL(https://sourceforge.net/projects/heimdall-astro/)。PRESTO是由Scott Ransom开发的大型脉冲星搜索和分析软件套件[9]，是开展脉冲星搜索最常用软件之一，该软件已参与发现超过700颗脉冲星（https://baijiahao.baidu.com/s?id=1689281702839604685&wfr=spider&for=pc）。FAST观测主要使用PRESTO进行脉冲星搜索[10]，但PRESTO开发较早，搜索流程大多采用单核串行方式，面对海量脉冲星观测数据，PRESTO搜索速度仍有欠缺，如果运用并行编程技术从而充分利用多核处理器计算

基金项目：国家自然科学基金（U1831131）；中国科学院天文大科学研究中心 FAST 重大成果培育项目（2019SR04）资助.
收稿日期：2022-10-26；修订日期：2022-11-29
作者简介：傅志明，男，硕士，研究方向：FAST 数据处理. Email: ernest0801@qq.com
***通信作者：**谢晓尧，男，教授，研究方向：网络通信、信息安全与人工智能. Email：xyx@gznu.edu.cn

资源，将明显提升数据处理性能，加快科学发现的步伐。本文首先介绍单脉冲搜索技术，分析当前单脉冲搜索流程及特点，对PRESTO中单脉冲搜索算法中的去趋势(detrend)算法进行优化，并对优化后的单脉冲搜索并行化。实验结果表明，算法优化及中央处理器并行化显著缩短了数据处理时间，单脉冲搜索性能得到一定的提升。

2 单脉冲搜索

单脉冲搜索算法最早由文[11]提出，综合影响所接收脉冲的信号强度、形状和宽度的各种因素，近似的信号模型描述为

$$I(t)=g_r g_d S(t)*h_{DM}(t)*h_d(t)*h_{Rx}(t)+N(t), \quad (1)$$

其中，源信号$S(t)$由对应于折射和衍射闪烁的调制因子g_r和g_d调制，同时卷积几个因素(用星号表示)，这些因素包括色散涂抹$h_{DM}(t)$、多路径传播引起的脉冲展宽$h_d(t)$以及接收器和数据采集系统的平均值$h_{Rx}(t)$，$N(t)$为附加接收机噪声，在很大程度上与$h_{DM}(t)$相关的色散涂抹可以从信号中去卷积。但如果没有使用精准的色散测量值(Dispersion Measure，DM)，可能会有残留的色散涂抹。

收集到的信号脉冲宽度描述为

$$w_{obs}=\left({t_{scatt}}^2+{t_{samp}}^2+{w_{int}}^2+{\Delta t_{DM}}^2+{\Delta t_{\delta DM}}^2\right)^{1/2}, \quad (2)$$

其中，t_{scatt}为散射时间；t_{samp}为采样时间；w_{int}为脉冲固有宽度；Δt_{DM}为由色散测量产生的随频率变化的涂抹，定义为

$$\Delta t_{DM}=8.3\frac{\Delta\nu DM}{\nu^3}(\mu s), \quad (3)$$

（3）式中，$\Delta\nu$为总带宽，ν为观测频率；$\Delta t_{\delta DM}$为使用不正确的消色散产生的涂抹，定义为

$$\Delta t_{\delta DM}=\frac{8.3}{4}\frac{N_{chan}\Delta\nu\delta DM}{\nu^3}(\mu s), \quad (4)$$

（4）式中，N_{chan}为频率通道数，δDM为试验色散测量值与真实色散测量值的误差。

根据文[11]提出的单脉冲搜索理论，PRESTO中单脉冲搜索主要分为3个阶段：数据准备、单脉冲搜索和结果诊断。单脉冲搜索流程如图1，其中，紫色部分为数据准备阶段，绿色部分为单脉冲搜索阶段，蓝色部分为结果诊断阶段。数据准备阶段主要是标记干扰信号，设定消色散方案并对观测数据消色散，得到不同色散测量值对应的dat文件；单脉冲搜索阶段主要对时间序列去趋势，匹配滤波，筛选候选体信息，根据候选体信息生成搜索结果图；结果诊断阶段主要对候选体信息进一步判断是否包含天体信号。

目前大量针对PRESTO软件的优化与改进工作取得了较好的效果，如文[12]使用图形处理器实现消色散加速，文[13]在中央处理器上实现消色散、快速傅里叶变换和加速搜索的并行化，文[14]开发的以单个观测数据文件划分任务的并行计算任务调度系统，文[15]针对FAST特定参数空间，优化了PRESTO脉冲星候选体图像输出，文[16]利用并行化技术在图形处理器上对PRESTO中的加速度搜索进行了优化。

针对PRESTO中单脉冲搜索阶段的优化，文[17]采用python中的multiprocessing模块实现基于中央处理器并行加速单脉冲搜索，但文中并未对具体实现及加速效果进行介绍，本文将对其具体介绍并进行实验复现。

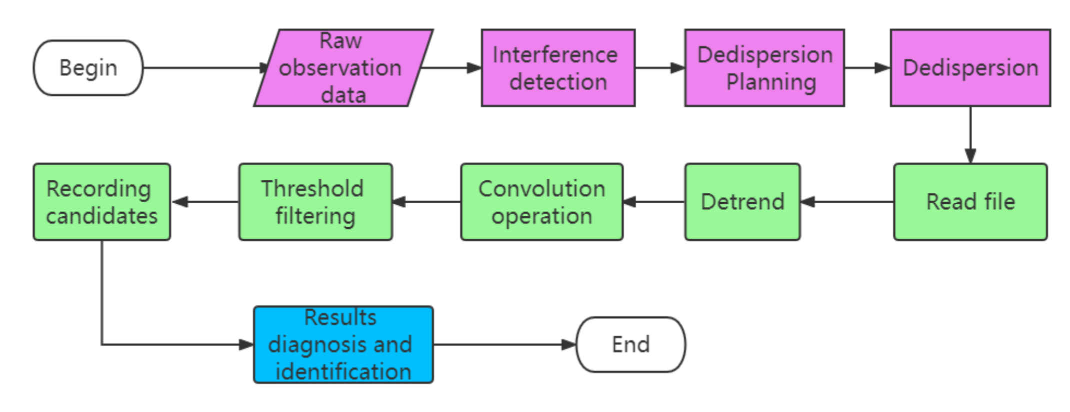


图 1 单脉冲搜索流程图

Fig. 1 Single pulse search process flowchart

3 单脉冲搜索算法优化及并行化

3.1 单脉冲搜索的算法优化

本文在处理器为AMD Ryzen 7 2700x eight-core processor*16，内存为64GB，操作系统为Ubuntu 18的硬件环境下，使用FAST的观测数据文件FH20201014_00C10.fits对PRESTO中单脉冲搜索程序single_pulse_search.py进行性能分析发现，去趋势算法部分约占single_pulse_search.py程序总时间开销的60%左右。single_pulse_search.py程序各步骤耗时情况如表1，如果能提高去趋势算法的性能，则PRESTO中单脉冲搜索算法的整体性能将明显提升。

表1 单脉冲搜索阶段各步骤耗时情况

Table 1 Time consumption of each step in the single pulse search process

	Read File	Detrend	Convolution Operation	Threshold Filtering	Recording Candidates	Others	Total Time Consumed
Consumed/s	0.61	92.65	18.01	9.31	0.61	28.07	149.28
Percentage/%	0.41	62.07	12.07	6.24	0.41	18.81	100.00

去趋势是抑制信号在采集过程中由于长时间观测导致的功率谱波动，以得到最佳检测效果。原程序中去趋势算法是调用Python数学计算库Scipy中信号处理模块signal中的detrend方法实现。该方法的核心是使用最小二乘法计算求得拟合直线的参数，但在该方法的运算流程中，存在重构以及复原数据形状，以及计算多余参数的计算，原detrend去趋势算法流程如图2(a)。本文对该方法进行优化，重新设计去趋势的运算流程，移除原方法中冗余计算，减少不必要开销，优化后去趋势算法流程如图2(b)：（1）原方法中数据形状重构及类型转换是为了得到适合计算的数据类型，在优化的算法中，直接将时间序列数据转换为Cython支持的list类型，减少数据形状重构及类型转换带来的运算开销。（2）原方法中最小二乘法分别计算回归系数(coef)、残差平方和(resids)、自变量秩(rank)和自变量奇异值(s)，而对于拟合直线，只需要计算回归系数即可；在优化算法中，重新设计最小二乘法只计算回归系数(coef)，并基于Cython编程方式实现最小二乘法，减少其他参数计算带来的开销。

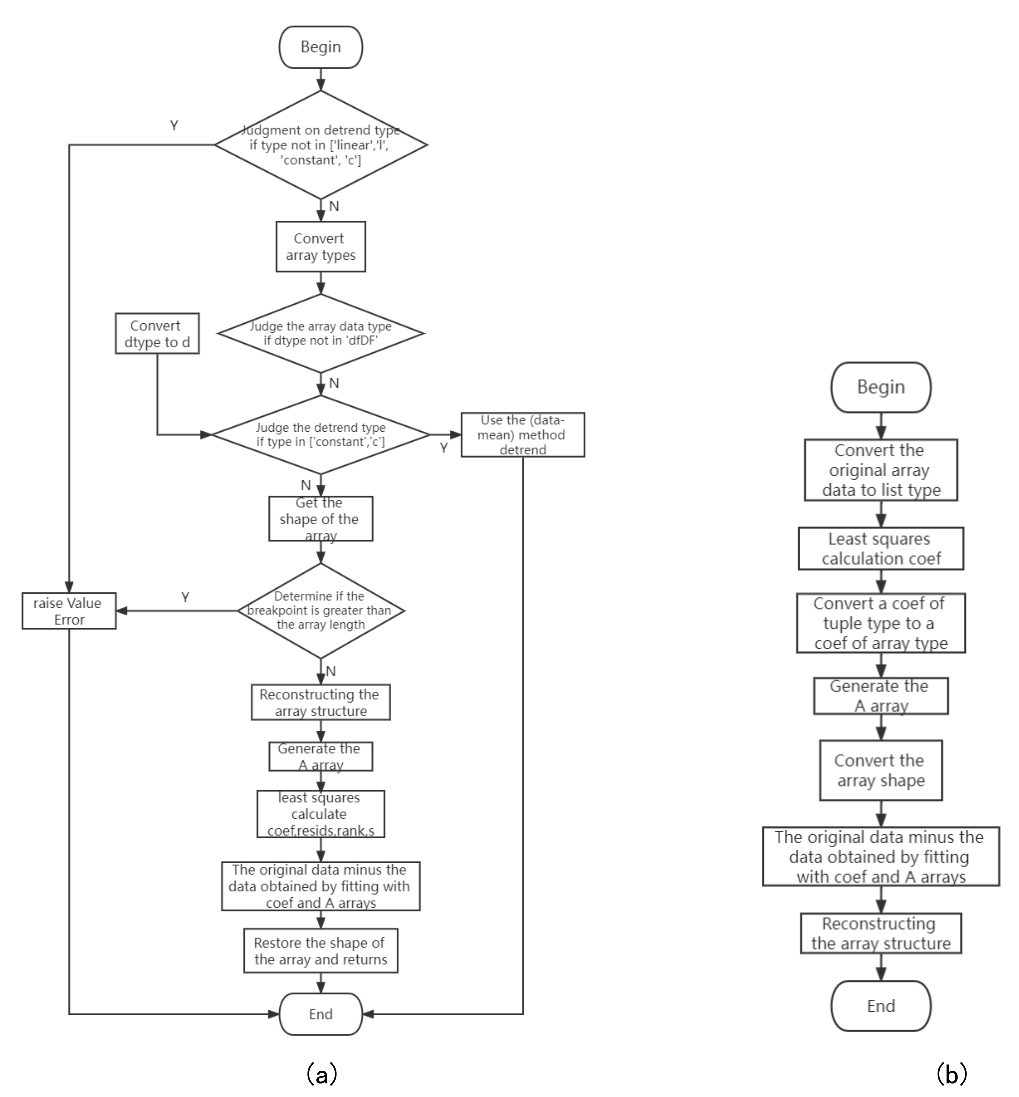


图 2 (a) detrend 方法运算流程图;(b) 优化后 detrend 运算流程图

Fig. 2 (a) Flow chart of detrend method;(b) flow chart of optimized detrend method

3.2 基于去趋势算法优化后的并行化

FAST脉冲星搜索场景需要处理大量小文件，同时伴随频繁的读写操作，如果基于图形处理器并行，面对大量小文件的搜索场景很难发挥图形处理器的高性能[18]，同时搜索算法本身存在过多数据依赖或控制依赖问题，这也极大影响加速效果，因此本文选择在中央处理器并行化。

理论上，一台计算机有多少个中央处理器内核，计算就可以同时运行多少个进程。而目前单脉冲搜索程序的算法还是基于单核串行设计的，这显然已经不符合多核时代并行算法主流，因此有必要从并行化角度出发，将串行执行的单脉冲搜索算法并行化，从而充分利用多核中央处理器的并行能力，提高程序的性能。

对单脉冲搜索流程分析，单脉冲搜索程序在对不同dat文件进行搜索时，输出不同singlepulse文件，不同dat文件之间不存在数据耦合的情况，每一个dat文件的搜索都相互独立，因此只需要给不同dat文件，使用不同进程各自进行搜索就可以实现单脉冲搜索并行化。

在原单脉冲搜索阶段的具体流程中，每一个dat文件得到的搜索结果，都需要写入一个全局列表中保存。如果对单脉冲搜索流程不加以改进便并行化，每个进程将搜索结果写入全局列表

会带来一部分进程间的通信开销，从而影响加速效果，原单脉冲搜索直接并行化示意图如图3(a)。为了减少进程间通信开销，对单脉冲搜索流程加以改进，将每一个dat文件的搜索结果返回，在所有dat文件搜索完成后，再将搜索结果写入全局列表中，改进后的单脉冲搜索并行化示意图如图3(b)。

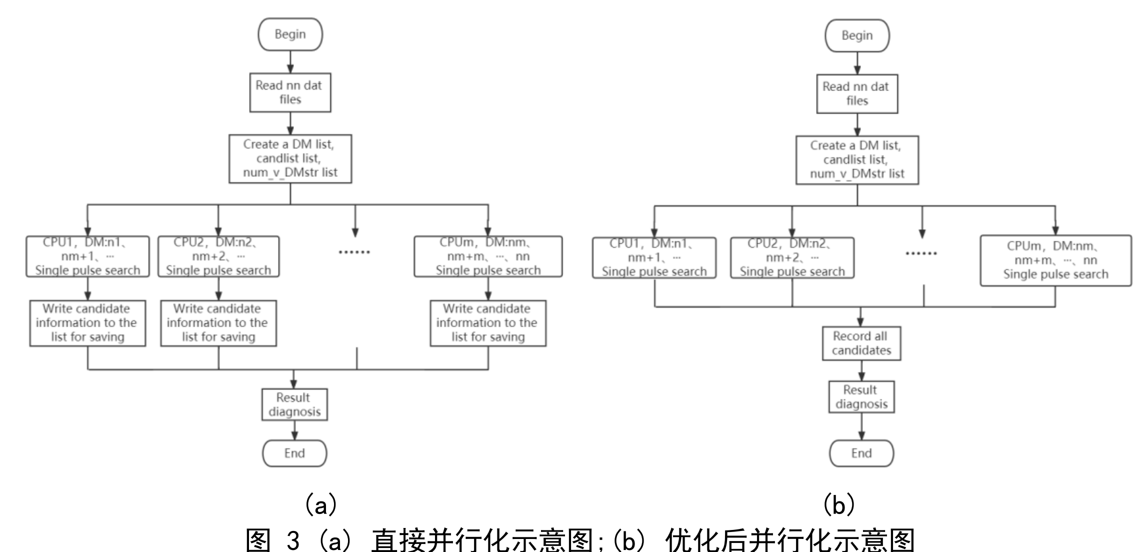


图 3 (a) 直接并行化示意图;(b) 优化后并行化示意图

Fig. 3 (a) Diagram of direct parallelization;(b) diagram of optimized parallelization

3.2.1 基于multiprocessing模块的并行化

Python标准库multiprocessing是一个多进程并行数据处理方案，可以帮助开发者轻松完成将程序从单进程执行转换为多进程并行执行，且技术复杂度不高。

对于不同的脉冲星观测文件，以及在DDplan规划消色散步长时使用的不同参数，生成的dat文件数量相差很大，如果并行程序在对每一个dat文件单脉冲搜索时都创建和销毁进程，会给程序运行带来一部分开销，从而影响程序整体性能。为此，在引入multiprocessing模块进行单脉冲搜索并行化的同时，使用进程池ProcessPoolExecutor管理多进程任务，以减少创建与销毁进程带来的开销，同时进程池还可以提高程序稳定性，避免程序因创建过多进程而导致程序崩溃。

使用multiprocessing模块并行化搜索时，首先设定进程池中进程的数量，由进程池管理这些dat文件的搜索任务。当有一个新的任务请求提交到进程池时，如果进程池中的并行任务未满，则创建一个新的进程执行新的任务请求；如果进程池中的并行任务已满，则该任务会等待，当进程池中有进程空闲时，便立即响应该任务请求。

3.2.2 基于Ray框架的并行化

上述multiprocessing模块实现的并行化，由于存在子进程与主进程之间的通信，导致中央处理器使用率不高，不能达到或者接近100%，对此，本文使用更加高性能的并行框架Ray[19]实现并行化。Ray是加州大学伯克利分校RISE实验室开发的分布式计算框架，可以灵活地运行任何计算密集型Python工作负载。

使用Ray对单脉冲搜索并行化示意图如图4，主进程执行时创建全局调度器及局部调度器，全局调度器在集群部署模式下才启用，在此不赘述。当主进程调用remote函数时，向局部调度器提交任务，再由局部调度器向本地机器中的worker进程分配任务，由worker进程完成计算并将结果返回。

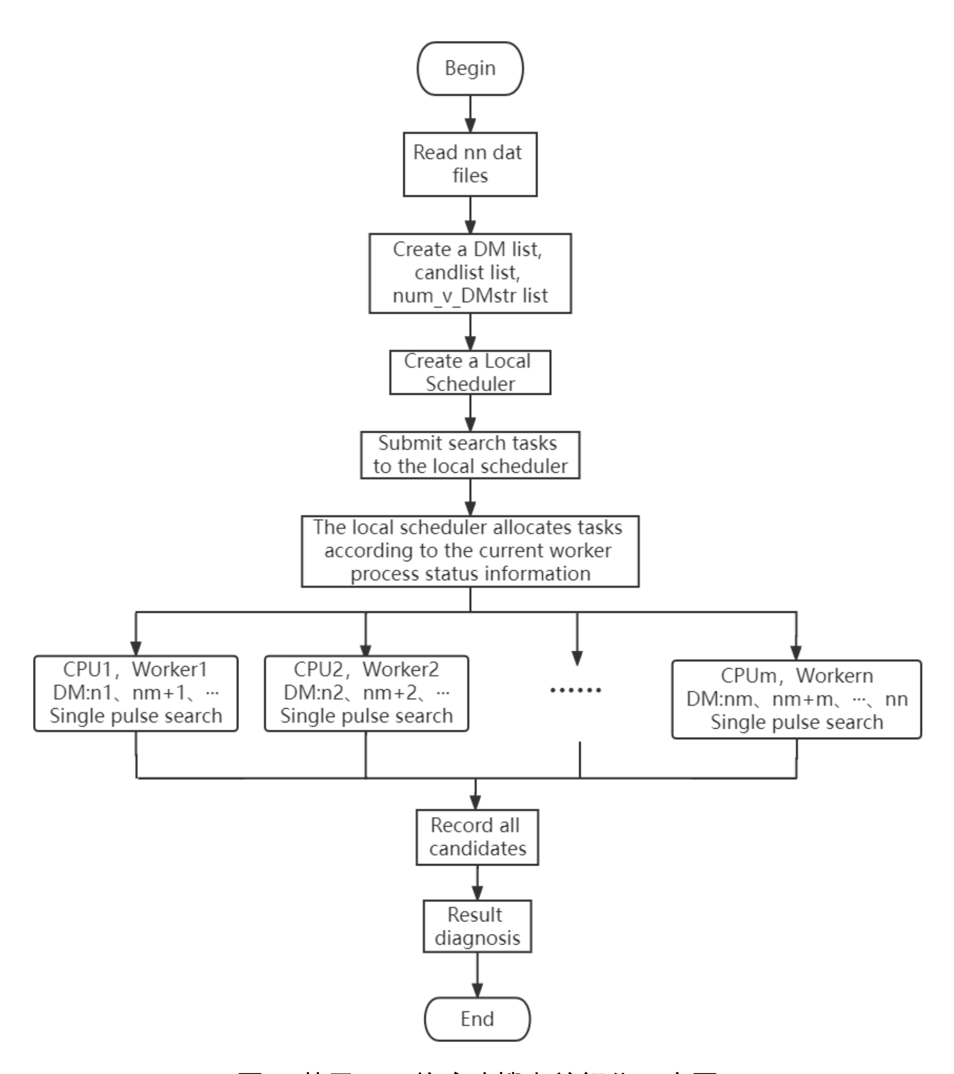


图 4 基于 Ray 单脉冲搜索并行化示意图

Fig. 4 Schematic diagram of parallelization of single pulse search based on Ray

3.3 实验结果分析

对单脉冲搜索程序算法优化及并行化对比实验所使用的硬件环境：处理器为 AMD Ryzen 7 2700x eight-core processor*16，内存为 64GB，操作系统为 Ubuntu 18。本文对 FAST 的观测数据文件(FH20201014_00C10，数据参数见表 2)消色散后生成的 3500 个 dat 文件进行测试。实验使用的数据参数情况如表 2，实验数据的消色散方案如表 3。

表 2 实验数据的参数

Table 2 Parameters of the experimental data

Parameter	Value	Parameter	Value
Sample time/μs	100	Spectra per file	262144
Central frequency/MHz	546	Time per subint(sec)	0.8192

Low channel/MHz	290.125	Time per file(sec)	26.2144
High channel/MHz	801.875	Bits per sample	8
Channel width/MHz	0.25	Bytes per spectra	2048
Number of channels	2048	Samples per subint	2048
Total Bandwidth/MHz	512	Bytes per subint	16777216
Spectra per subint	8192	Samples per subint	16777216

表 3 实验数据的消色散方案

Table 3 Dedispersion scheme for experimental data

Low DM	High DM	dDM	DownSamp
0.0	50.0	0.05	1
50.0	100.0	0.05	2
100.0	175.0	0.05	2

在相同的实验环境以及实验数据时，分别运行优化detrend算法前后的单脉冲搜索程序10次，记录每次运行时间，得到图5的实验结果，横轴为运行次数，纵轴为耗时（单位：s），single_pulse_search_raw为原程序耗时情况折线图，single_pulse_search_rm_rad为原程序去除冗余计算后耗时情况折线图，single_pulse_search_Cy_raw为使用Cython重新编程原detrend算法后耗时情况折线图，single_pulse_search_Cy_rm_rad为重新设计detrend算法并使用Cython编程优化后耗时情况折线图。由图5可以看出，重新设计detrend算法并使用Cython编程优化，相比于原程序性能提升约1.8倍。

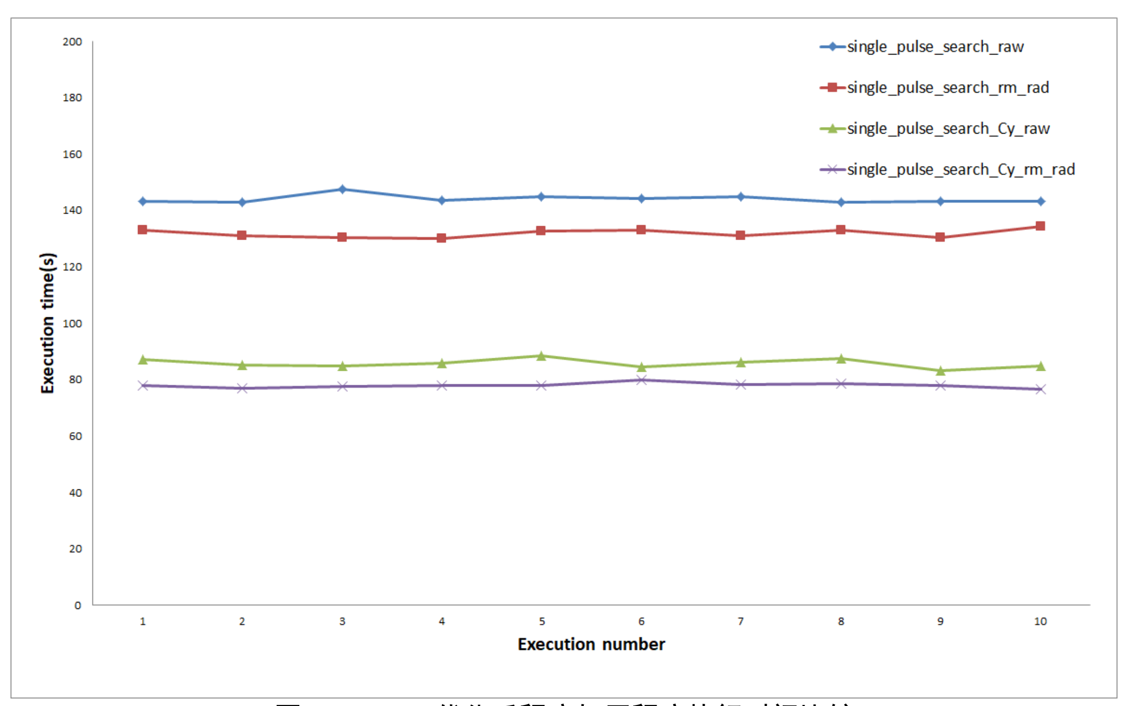


图 5 Cython 优化后程序与原程序执行时间比较

Fig. 5 Comparison of time consumed between Cython optimized program and original program

在使用Cython编程优化detrend算法基础上，使用Ray框架实现并行化，分别运行原单脉冲搜索程序、基于multiprocessing并行化程序及基于Ray并行化程序10次，记录每次运行时间，得到图6的实验结果，横轴为运行次数，纵轴为耗时（单位：s），raw为原程序耗时折线图，Cy_detrend

为detrend算法优化后耗时折线图，multiprocessing为基于detrend算法优化并使用multiprocessing模块对程序中央处理器并行化后耗时折线图，Ray为基于detrend算法优化并使用Ray框架对程序中央处理器并行化后耗时折线图。由图6可知，采用Ray框架对单脉冲搜索程序改进后，本文并行化处理方法耗时相比原单脉冲搜索程序有极大提升，加速比提高约10倍，较multiprocessing模块并行化版本也有一定进步，提高约3倍。综合来看，对程序的优化及并行化有一定的性能提升效果。

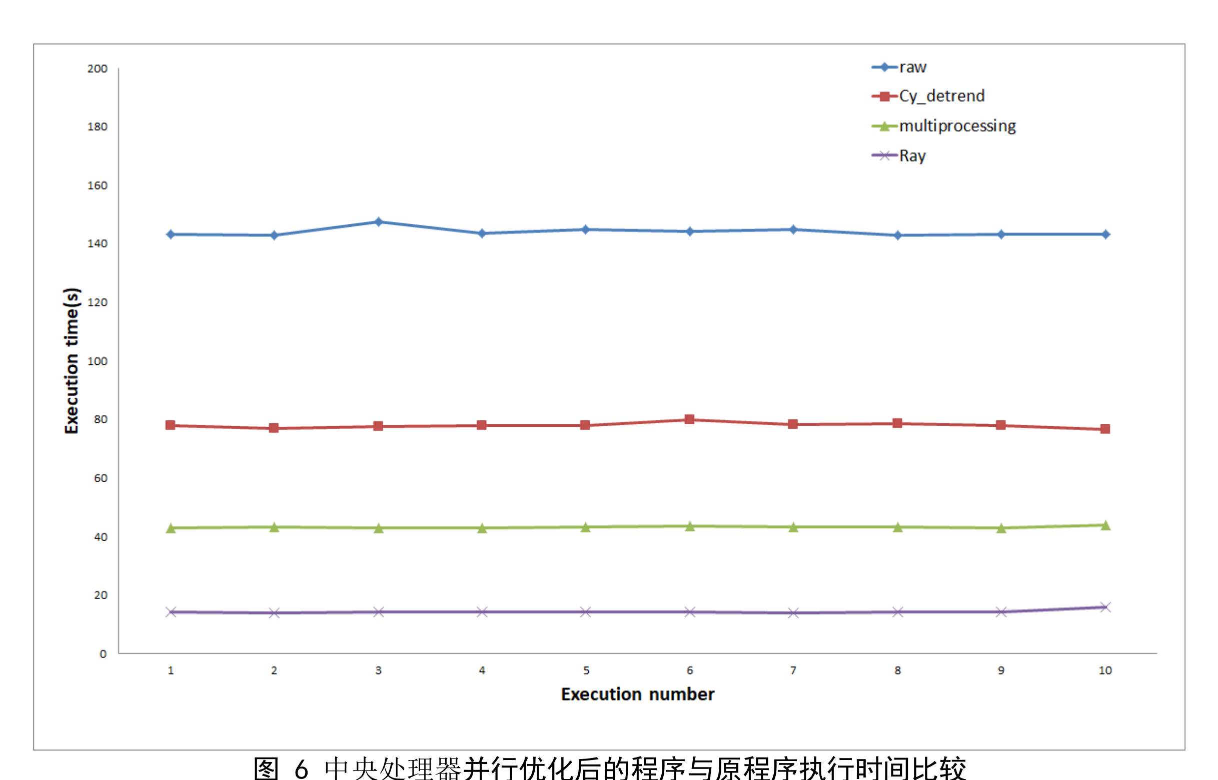


图 6 中央处理器并行优化后的程序与原程序执行时间比较

Fig.6 Comparison of CPU parallel optimized program and original program execution time

对比优化前后生成的搜索结果图可以看出，本文提出的基于去趋势算法优化的并行化单脉冲搜索算法能够保持串行算法的搜索效果，没有因算法优化及并行化而出现搜索结果不一致的情况。程序优化前后的搜索结果图如图7，(a)为原程序串行生成的搜索结果图，(b)为detrend算法优化后生成的搜索结果图，(c)为使用multiprocessing模块并行化生成的搜索结果图，(d)为使用Ray框架并行化生成的搜索结果图。

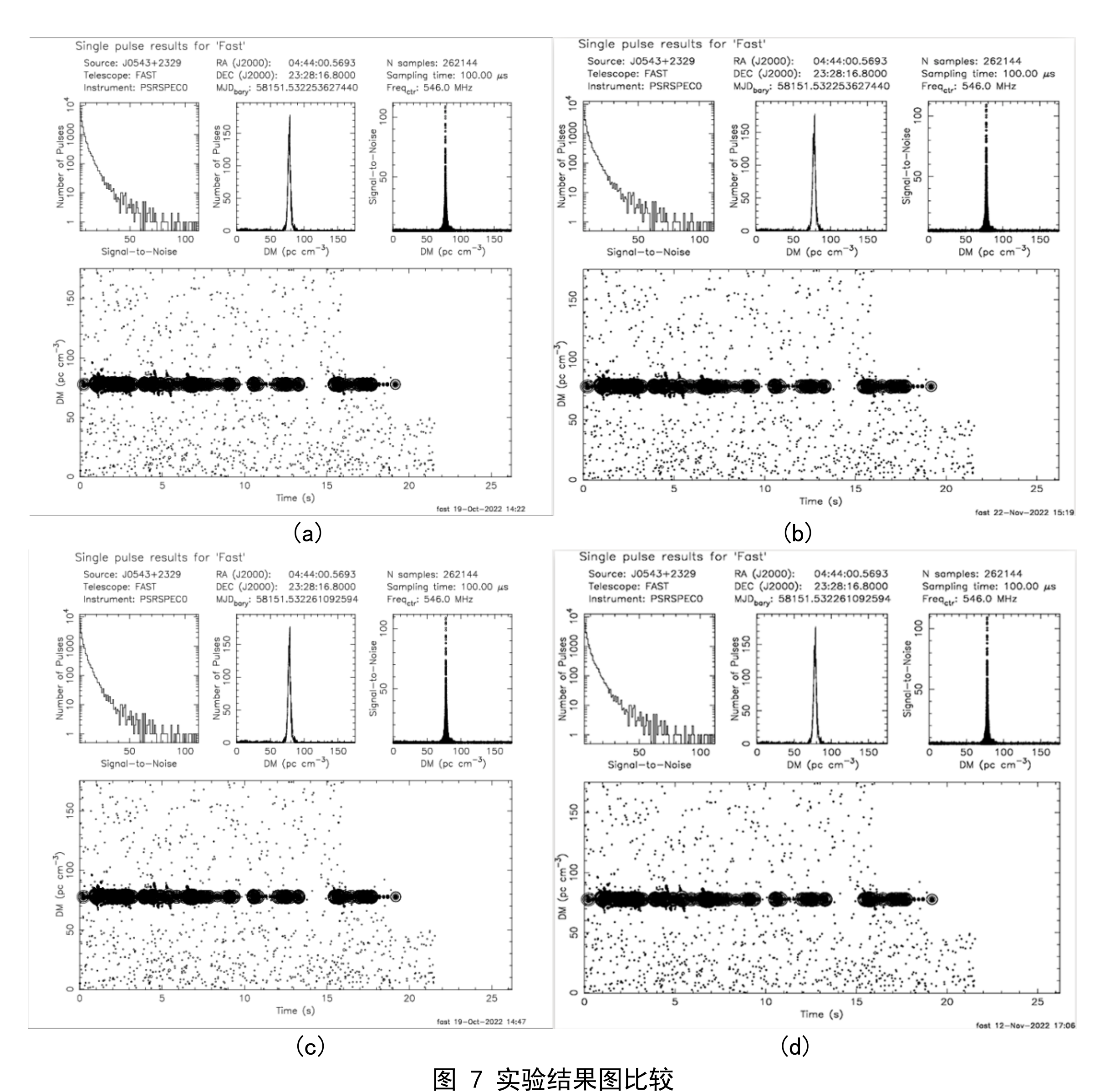


图 7 实验结果图比较

Fig. 7 Comparison of experimental results graphs

4 总结与展望

本文对PRESTO中单脉冲搜索技术进行了介绍及分析，基于单脉冲搜索技术的特点，分别使用Cython编程、multiprocessing并行模块、Ray并行框架对PRESTO中原单脉冲搜索程序进行优化，在保证搜索效果的基础上，显著提升了单脉冲搜索程序性能。本文算法优化及并行化的工作都是基于纯中央处理器环境，无需依赖CUDA和图形处理器，无需修改代码，即可实现在中央处理器环境下高性能单脉冲搜索。

原单脉冲搜索程序是基于python开发的，重新基于C或C++语言实现程序能够达到更好的加速效果，但开发周期很长，为了进一步提升单脉冲搜索程序性能，下一步可以尝试将PRESTO中的单脉冲搜索程序完全使用C或C++语言实现，并使用Ray框架搭建集群部署进行分布式并行计算，以更好地满足日益增长的脉冲星观测数据的快速搜索。

致谢：本文在FAST（500m口径球面射电望远镜）数据基础上完成。FAST是由中国科学院国家天文台运行和管理的国家大科学装置。感谢贵州省信息与计算科学重点实验室提供数据支持。

Algorithm Optimization and Parallelization of PRESTO Single Pulse Search

Fu Zhiming[1], Chen Zonghao[1], Liang Nan[1], Yu Xuhong[1], Shao Guangsheng[2], Zhang Bin[1], Xie Xiaoyao[1*]

(1.Guizhou Provincial Key Laboratory of Information and Computing Science, Guizhou Normal University, Guiyang 550001 China，Email：xyx@gznu.edu.cn;

2.Guizhou Normal University Library, Guizhou Normal University, Guiyang 550001 China)

Abstract: Since the discovery of some objects without obvious periodicity, such as fast radio bursts and rotating radio temporary sources, the single pulse search approach has attracted extensive attention of researchers. Meanwhile, the radio telescope equipment is becoming more and more perfect, which brings higher resolution and wider observation space. Therefore, the amount of data generated by pulsar observation increases dramatically, and it is urgent to speed up the data processing of pulsars. The single pulse search in PRESTO (PulsaR Exploration and Search TOolkit) is introduced, the detrending algorithm in single pulse search is optimized by Cython programming, and the parallelization of single pulse search on CPU (Central Processing Units) is realized by Ray Framework. The experimental results show that the parallelization of the optimized single pulse search algorithm can significantly improve the performance of the search program and shorten the data processing time. At the same time, the parallel strategy is only based on CPU, so it can realize high-performance data processing in pure CPU environment without modifying the code.

Key words: single pulse search; pulsar; parallelization

参考文献:

［1］ STAELIN D H. Fast folding algorithm for detection of periodic pulse trains[J]. Proceedings of the IEEE, 1969, 57(4): 724-725.

［2］ BACKER D C. Pulsar nulling phenomena[J]. Nature, 1970, 228(5266): 42-43.

［3］WANG N, MANCHESTER R N, JOHNSTON S. Pulsar nulling and mode changing[J]. Monthly Notices of the Royal Astronomical Society, 2007, 377(3): 1383-1392.

［4］ LORIMER D R, BAILES M, MCLAUGHLIN M A, et al. A bright millisecond radio burst of extragalactic origin[J]. Science, 2007, 318(5851): 777-780.

［5］ MCLAUGHLIN M A, LYNE A G, LORIMER D R, et al. Transient radio bursts from rotating neutron stars[J]. Nature, 2006, 439(7078): 817-820.

［6］NORRIS R P. Data challenges for next-generation radio telescopes[C]// Proceedings of the Sixth IEEE International Conference on e-science Workshops. 2011:21-24.

［7］ LI D, WANG P, QIAN L, et al. FAST in space: considerations for a multibeam, multipurpose survey using China's 500-m aperture spherical radio telescope (FAST)[J]. IEEE Microwave Magazine, 2018, 19(3): 112-119.

［8］ SWIGGUM J K, LORIMER D R, MCLAUGHLIN M A, et al. Arecibo pulsar survey using ALFA. III. precursor survey and population synthesis[J]. The Astrophysical Journal, 2014, 787(2): 137.

［9］ RANSOM S M. New search techniques for binary pulsars[C]// Bulletin of the American Astronomical Society. 2001: 99-102.

［10］ 潘之辰, 钱磊, 岳友岭. 脉冲星搜索技术及 FAST 望远镜脉冲星搜索展望[J].天文研究与技术, 2017, 14(1): 8-16

PAN Z C, QIAN L, YUE Y L. Pulsar searching techniques and their applications to FAST pulsar search[J]. Astronomical Research & Technology, 2017, 14(1): 8-16.

［11］CORDES J M, MCLAUGHLIN M A. Searches for fast radio transients[J]. The Astrophysical Journal, 2003, 596(2): 1142.

［12］YOU S P, WANG P, YU X H, et al. A GPU based single-pulse search pipeline (GSP) with database and its application to the Commensal Radio Astronomy FAST Survey (CRAFTS)[J]. Research in Astronomy and Astrophysics, 2022, 21(12): 314.

［13］ YU Q Y, PAN Z C, QIAN L, et al. A PRESTO-based parallel pulsar search pipeline used for FAST drift scan data[J]. Research in Astronomy and Astrophysics, 2020, 20(6): 091.

［14］张辉,谢晓尧,李菂, 等.一种面向FAST PB量级脉冲星数据处理加速方法及系统[J].天文研究与技术,2021,18(1):129-137

ZHANG H, XIE X Y, LI D, et al. A data processing acceleration method and system for FAST Petabyte pulsar data processing[J]. Astronomical Research & Technology, 2021, 18(1): 129-137.

［15］ 张翔,刘志杰,王培,等. PRESTO搜寻脉冲星中候选体识别效率的提高[J].贵州师范大学学报(自然科学版),2018,36(3):84-88.

ZHANG X, LIU Z J, WANG P, et al. Improvement of candidate recognition accuracy in PRESTO pulsar search[J]. Journal of Guizhou Normal University (Natural Sciences), 2018, 36(3):84-88.

［16］ 冀昶旭,于徐红,刘志杰.脉冲双星加速度搜索方法及软件改进[J].天文研究与技术,2022,19(2):103-110.

JI C X,YU X H, LIU Z J. Binary pulsar system acceleration search method and software

improvement[J]. Astronomical Research & Technology, 2022, 19(02):103-110.

［17］游善平. 脉冲星搜索数据处理的并行加速研究及应用[D]. 贵州: 贵州大学, 2021.

YOU S P. Research and application of parallel acceleration of pulsar search data processing[D]. Guizhou: Guizhou University,2021.

［18］ YU Q Y, PAN Z C, QIAN L, et al. A PRESTO-based parallel pulsar search pipeline used for FAST drift scan data[J]. Research in Astronomy and Astrophysics, 2020,20 (6): 091.

［19］ MORITZ P, NISHIHARA R, WANG S, et al. Ray: a distributed framework for emerging {AI} applications[C]//Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). 2018: 561-577.